AFRL-RI-RS-TR-2015-061

ADVANCED MULTIPLE IN-MULTIPLE OUT (MIMO) ANTENNA COMMUNICATIONS FOR AIRBORNE NETWORKS

SYRACUSE UNIVERSITY

MARCH 2015

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2015-061 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /
PAUL J. OLESKI
Work Unit Manager

/ S /
MARK H. LINDERMAN
Technical Advisor, Computing
& Communications Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MAR 2015	FINAL TECHNICAL REPORT	OCT 2011 – SEP 2014

4. TITLE AND SUBTITLE

ADVANCED MULTIPLE IN-MULTIPLE OUT (MIMO) ANTENNA COMMUNICATIONS FOR AIRBORNE NETWORKS

5a. CONTRACT NUMBER
FA8750-11-1-0040

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Biao Chen

5d. PROJECT NUMBER
MIMO

5e. TASK NUMBER
TE

5f. WORK UNIT NUMBER
CH

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Syracuse University
900 South Crouse Ave
Syracuse NY 13244

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RITE
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2015-061

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The objective of this project was to study and implement a MIMO architecture in airborne communications. The primary challenges include: 1) the lack of scatters in airborne communications which often renders the channel matrices to be rank deficient; 2) fast channel variation due to high mobility of the communication platforms. Toward overcoming these challenges and in close collaboration with AFRL/RITE researchers, we have:
- explored the feasibility of channel tracking in a dynamic channel environment
- developed a GUI system that allows one to visualize the effective capacity under various communications scenarios
- developed a D-BLAST architecture that employs orthogonal spreading to counter the effect of channel rank deficiency
- implemented, using python and with four USRP N210 nodes, a 2x2 MIMO system with over the air transmission.

15. SUBJECT TERMS
Multiple In-Multiple Out (MIMO Antenna Communications, Airborne Networks, D-BLAST, channel tracking, USRP N210 nodes

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**PAUL J. OLESKI**
U	U	U	UU	66	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

Contents

List of Figures

List of Tables

Summary

This report summarizes the results of the research and development effort under the auspice of AFRL Award ♯FA8750-11-1-0040. The primary objective of the project is to study and implement MIMO architecture in airborne communications. The primary challenges include 1) the lack of scatterers in airborne communications which often renders the channel matrices to be rank deficient; 2) fast channel variation due to high mobility of the communication platforms.

Toward overcoming these challenges and in close collaboration with AFRL researchers, we have

- explored the feasibility of channel tracking in a dynamic channel environment;
- developed a GUI system that allows one to visualize the effective capacity under various communication scenarios;
- developed a D-BLAST architecture that employs orthogonal spreading to counter the effect of channel rank deficiency;
- implemented, using python and with four USRP N210 [1] nodes, a 2×2 MIMO system with over the air transmission.

The following refereed articles have been published under this effort:

- K. Borle, B. Chen, and M. J. Gans, Channel tracking for D-BLAST for airborne platforms, *Proc. Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, Nov. 2011
- M. J. Gans, K. Borle, B. Chen, T. Freeland, D. McCarthy, R. Nelson, D. Overrocker and P. Oleski, Enhancing connectivity of unmanned vehicles through MIMO communications, *Proc. of IEEE Vehicular Technology Conference, Las Vegas*, NV, Sept. 2013.

In addition, various software packages have been delivered to the AFRL collaborators, including

- Channel tracker simulator, as described in Chapter 3.

- Channel matrix simulator. This is to take the Newport measurement data and construct a new channel matrix according to the relative position of the transmitter and receiver through extrapolation.
- LDPC encoder and decoder source code.
- C++ code for the MIMO D-BLAST transmitter and receiver blocks.

Introduction

A point to point airborne MIMO communication link can be defined as two terminals communicating over a wireless channel such that either one or both the terminals are airborne and both employ multiple antennas. On the other hand, the conventionally studied MIMO wireless communication is based on the premise that the communicating multi-antenna terminals are located in a dense urban environment. Two important factors distinguish such an airborne MIMO channel from a conventional MIMO channel: Firstly, the channel conditions change very rapidly in an airborne link due to fast relative motion between the airborne terminals. Secondly, the channel offers a significantly less scattering environment as compared to a dense urban environment. As such, airborne MIMO communication poses a somewhat different set of problems as compared to the conventional MIMO communication problems.

This report describes our efforts in addressing the above mentioned problems of airborne MIMO communication. To be precise we address the problems of channel tracking in dynamic channel conditions and communication over rank deficient channels. In addressing these problems, we have encountered D-BLAST transceiver architecture as the central idea, upon which our proposed solutions are based. Hence, to facilitate experiments, we also develop a GNU Radio/USRP based D-BLAST communication system employing 2 transmit and 2 receive antennas. Furthermore we have developed a C++ library that provides D-BLAST encoding and decoding functionality. The C++ library is not coupled to the GNU Radio framework and hence provides more flexibility in its usage.

In the following we provide a brief description of the contributions followed by the organization of the report.

2.1 Channel Tracking

The channel in airborne communication changes continuously at a rapid rate. To facilitate coherent symbol detection in such a rapidly changing channel, the receiver needs to update the channel state information at a much more frequent rate. The traditional use of embedding pilot symbols with payload data may not be easy to justify if channel variation is fast and/or data rate requirement is high. We consider the implementation of D-BLAST for airborne platforms and develop channel tracking scheme to eliminate or reduce the use of pilot symbols. In a normal operation mode, channel update is achieved dynamically as

each layer of the D-BLAST, encoded using an LDPC code, is decoded. To ensure that the transceiver can detect outage due to the loss of channel state, an adaptive algorithm is devised utilizing the extremal property of terminating likelihood ratio of an LDPC decoder.

2.2 Variable Rate MIMO

In this part of the report we develop a variable rate MIMO scheme, based on D-BLAST transceiver architecture, to overcome MIMO channels that are rank deficient. Real channel measurements are conducted and the analysis supports the use of MIMO communications for such applications due to its potential throughput advantage. Unique challenges and the ways to address them are described in detail. In particular, the lack of scattering and the blockage of line of sight may lead to rank deficient channel matrices, which are exacerbated due to the absence of channel state information at the transmitter. A variable rate MIMO scheme is thus proposed to overcome these challenges in order to realize the promising throughput gain afforded by MIMO communication.

2.3 GNU Radio MIMO Implementation Using USRP N210

An indispensable part of any scientific inquiry is its experimental component. The aforementioned Variable Rate MIMO scheme shows promising results for the rank deficient channels obtained through real measurement data. The results, as encouraging as they appear, nonetheless must be validated through experimental means. Therefore, to conduct real-time wireless experiments we need a platform that facilitates efficient and fast prototyping of a testbed. Software defined radio, which provides us with these desirable traits, appears as a very attractive option for developing the necessary components required for the experimentation.

The GNU Radio software development toolkit combined with USRPs provide an efficient and rapidly employable software defined radio system. Using the GNU Radio/USRP platform, we have developed a 2×2 D-BLAST transceiver. We have also created a C++ library, developed separately from the GNU Radio based testing framework but able to be used within the framework.

2.4 Organization

The rest of the report is organized as follows.

- Chapter 3 develops and analyzes the channel tracking scheme for a D-BLAST based airborne communication system.
- Chapter 4 develops a variable rate MIMO scheme, based on D-BLAST transceiver architecture, to overcome MIMO channels that are rank deficient.

- Chapter 5 provides a brief overview of the C++ based software library which is developed to facilitate experimentation of MIMO communication.
- Chapter 6 concludes the report.

Channel Tracking

3.1 Introduction

For a communication system in which either one or both the stations are in motion, the channel is continuously changing. At the receiver end, for coherent detection of symbols, this translates to constant updating of channel state information. The conventional method is to multiplex pilots and data symbols. For the case where there is rapid variation in channel, this pilot overhead may not be affordable. In order to tackle this scenario we try to investigate the case in which we use previously detected symbols to estimate channel for next block.

Since, the channel varies temporally, simply using the previous block of decoded symbols may not be optimal. We try to solve this problem by first finding the optimal length of symbols needed to minimize the mean square error of estimated channel for a SISO channel with rayleigh fading. Then we generalize this to MIMO case.

Once we know how to track the channel efficiently, the question arises, how to detect outage at the receiver. We narrow down our search for an outage detection scheme to systems which use LDPC codes. We find through simulations that the average mean magnitude of the log liklihood ratios remain small when the codeword is docoded incorrectly, while in the otherwise case, the mean magnitude has a high value.

Section 3.2.1 gives the analysis for optimal length to minimize the mean squared error in a Rayleigh fading channel. We then give a brief description of the D-BLAST architecture in Section 3.2.2. Section 3.2.3 describes the channel tracking algorithm for D-BLAST, followed by simulation results. Section 3.4 briefly describes the channel tracking simulator, which we have developed.

3.2 Methods, Assumptions and Procedures

3.2.1 Channel Estimation - Optimal Length

Single Input Single Output Case

Consider a SISO channel, $y_i = h_i x_i + z_i$, where x_i is the transmitted symbol, z_i is the additive noise and h_i is the fading coefficient in the i^{th} time slot [2]. If the transmitter transmits $N+1$

symbols in $N+1$ consecutive time slots,

$$\begin{bmatrix} y_N \\ y_{N-1} \\ \vdots \\ y_0 \end{bmatrix} = \begin{bmatrix} h_N & 0 & \dots & 0 \\ 0 & h_{N-1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & h_1 0 \end{bmatrix} \begin{bmatrix} x_N \\ x_{N-1} \\ \vdots \\ x_0 \end{bmatrix} + \begin{bmatrix} z_N \\ z_{N-1} \\ \vdots \\ z_0 \end{bmatrix}. \tag{1}$$

Write the diagonal matrix of fading coefficients as $h_{N-k} = h_N + \Delta_k$, for $k = 1, 2, \dots N$ [3], eq. (1) becomes

$$\begin{bmatrix} y_N \\ y_{N-1} \\ \vdots \\ y_0 \end{bmatrix} = h_N \mathbf{I}_N \mathbf{x} + \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & \Delta_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Delta_N \end{bmatrix} \begin{bmatrix} x_N \\ x_{N-1} \\ \vdots \\ x_0 \end{bmatrix} + \begin{bmatrix} z_N \\ z_{N-1} \\ \vdots \\ z_0 \end{bmatrix}, \tag{2}$$

where $\mathbf{I}_N$ is an $N \times N$ identity matrix and $\mathbf{x} = [x_N \; x_{N-1} \dots x_1]^T$ Multiplying both sides by $\mathbf{x}^H$ we get,

$$\mathbf{x}^H \mathbf{y} = h_N \mathbf{x}^H \mathbf{x} + \sum_{k=1}^{N} \Delta_k x_k x_k^* + \sum_{k=0}^{N} x_k^* z_k. \tag{3}$$

For simplicity, let $x_k x_k^* = 1$. The linear least squares estimate is given by,

$$\begin{aligned} \tilde{h}_N &= \frac{1}{N+1} \mathbf{x}^H \mathbf{y} \\ &= h_N + \frac{1}{N+1} \sum_{k=1}^{N} \Delta_k + \frac{1}{N+1} \sum_{k=0}^{N} x_k^* z_k. \end{aligned} \tag{4}$$

The error in estimating h_N thus arises from two factors: One due to temporal channel variation and the other due to additive noise component. Our goal is to find the optimum training length that provides the best trade-off between these two error types. This is equivalent to minimizing the average of $|\tilde{h}_N - h_N|^2$. To achieve this we first try to find a tractable form of the MSE,

$$\begin{aligned} &\mathscr{E}|\tilde{h}_N - h_N|^2 \\ &= \mathscr{E}\left[\left(\tilde{h}_N - h_N\right)\left(\tilde{h}_N - h_N\right)^*\right] &(5) \\ &= \frac{1}{(N+1)^2} \mathscr{E}\left[\left(\sum_{k=1}^{N} \Delta_k\right)\left(\sum_{k=1}^{N} \Delta_k^*\right)\right] \; + \frac{\sigma_z^2}{N+1}, &(6) \end{aligned}$$

where σ_z^2 is the noise variance. The expectation of the cross-product terms is 0 because we assume that the noise is zero mean and is independent of channel coefficients. To evaluate $\mathscr{E}|\tilde{h}_N - h_N|^2$, we first evaluate the expectation in the first term in (6). We use the approach similar to [4]. We assume the channel as frequency flat fading with a diffuse scattering channel model, which lends itself mathematical tractability. The following theorem summarizes our result.

Theorem 1. *Define* $\Delta_k = h((N-k)T_s) - h(NT_s)$. $h(t)$ *is the channel modeled as follows,*

$$h(t) = \sum_{k=1}^{S} \beta_k e^{j2\pi f_k t}, \tag{7}$$

where S *is the number of scatterers,* β_k *and* f_k *are the complex amplitude and Doppler frequency of the* k^{th} *multipath respectively and* T_s *is the symbol period, with the following assumptions.*

1. β_k *are independent zero mean random variables, and normalized such that* $\sum_{k=1}^{S} |\beta_k|^2 = 1$.
2. f_k *are i.i.d. distributed uniformly on the interval* $[-f_D, f_D]$.
3. β_k *and* f_k *are independent of each other.*

Then, (See Appendix A)

$$\begin{aligned} &\mathscr{E}\left[\left(\sum_{k=1}^{N} \Delta_k\right)\left(\sum_{k=1}^{N} \Delta_k^*\right)\right] \\ &= N(N+1) - \frac{\sin(\pi f_D N T_s)\sin(\pi f_D (N+1) T_s)}{\pi f_D T_s \sin(\pi f_D T_s)}. \end{aligned} \tag{8}$$

Therefore, putting (8) in (6), we get

$$\begin{aligned} \mathscr{E}|\tilde{h}_N - h_N|^2 = \frac{\sigma_z^2}{N+1} + \frac{N}{N+1} \\ - \frac{\sin(\pi f_D N T_s)\sin(\pi f_D (N+1) T_s)}{(N+1)^2 \pi f_D T_s \sin(\pi f_D T_s)}. \end{aligned} \tag{9}$$

The above expression gives the MSE as a function of maximum Doppler spread, symbol period and number of training symbols. For a given Doppler spread and symbol time, we can find the optimal training length numerically. We consider the MIMO case in the following section.

A plot of MSE against the fraction of code block size used for estimation is given in Fig. 1 for $f_D T_s = 1.25 \times 10^{-4}$, where f_D is the maximum Doppler spread and T_s is the symbol period.

Multiple Input Multiple Output Case

Consider a MIMO system with t transmit antennas and r receive antennas. We assume the following channel model, $\mathbf{y}_i = \mathbf{H}_i \mathbf{x}_i + \mathbf{z}_i$, where $\mathbf{x}_i$ is the $t \times 1$ transmitted vector, $\mathbf{y}_i$ is the $r \times 1$ received vector, $\mathbf{H}_i$ is the $r \times t$ channel matrix and $\mathbf{z}_i$ is the $r \times 1$ noise vector at time index i. We assume the noise to be complex Gaussian with zero mean and covariance matrix equal to the identity matrix.

Let the transmitter send $N+1$ symbol vectors in $N+1$ time slots. The $N+1$ transmitted symbols from one antenna are orthogonal to that of other transmit antennas.

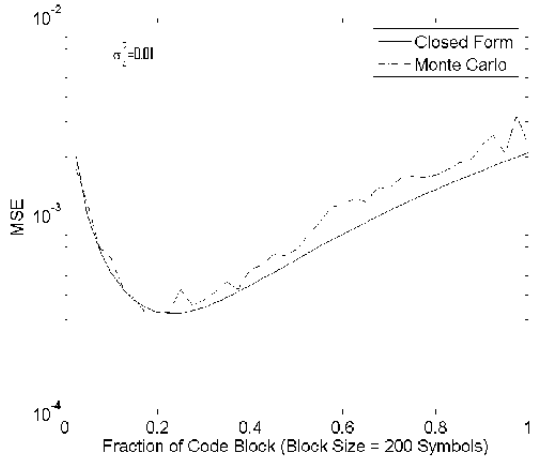


Figure 1: Behavior of MSE of SISO channel for $f_D T_s = 1.25 \times 10^{-4}$ with block length of 200 symbols

$$\mathbf{y_N} = \mathbf{H_N x_N} + \mathbf{z_N}, \tag{10}$$

$$\begin{aligned}\mathbf{y_{N-1}} &= \mathbf{H_{N-1} x_{N-1}} + \mathbf{z_{N-1}} \\ &= \mathbf{H_N x_{N-1}} + \mathbf{\Delta_1 x_{N-1}} + \mathbf{z_{N-1}},\end{aligned} \tag{11}$$

$$\vdots$$

$$\begin{aligned}\mathbf{y_0} &= \mathbf{H_0 x_0} + \mathbf{z_0} \\ &= \mathbf{H_N x_0} + \mathbf{\Delta_N x_0} + \mathbf{z_0}.\end{aligned} \tag{12}$$

The MSE in estimating $\tilde{\mathbf{H}}_N$ can be given as follows,

$$\mathscr{E}||\tilde{\mathbf{H}}_N - \mathbf{H}_N||_F^2 = \mathscr{E} \sum_{i,j} \left|\tilde{h}_N^{(i,j)} - h_N^{(i,j)}\right|^2 \tag{13}$$

$$= \sum_{i,j} \mathscr{E} \left|\tilde{h}_N^{(i,j)} - h_N^{(i,j)}\right|^2 \tag{14}$$

$$= rt\ \mathscr{E} \left|\tilde{h}_N^{(1,1)} - h_N^{(1,1)}\right|^2. \tag{15}$$

$h_N^{(i,j)}$ is the channel coefficient between the i^{th} receiver antenna and the j^{th} transmitter antenna of $\mathbf{H_N}$. The third equality is because all the elements are assumed i.i.d. The expectation in (15) can be reduced to that in expression (9).

Therefore, we can write the MSE as

$$\begin{aligned}&\mathscr{E}||\tilde{\mathbf{H}}_N - \mathbf{H}_N||_F^2 \\ &= rt \left(\frac{\sigma_z^2}{N+1} + \frac{N}{N+1} \right. \\ &\left. - \frac{\sin(\pi f_D N T_s) \sin(\pi f_D (N+1) T_s)}{(N+1)^2 \pi f_D T_s \sin(\pi f_D T_s)} \right).\end{aligned} \tag{16}$$

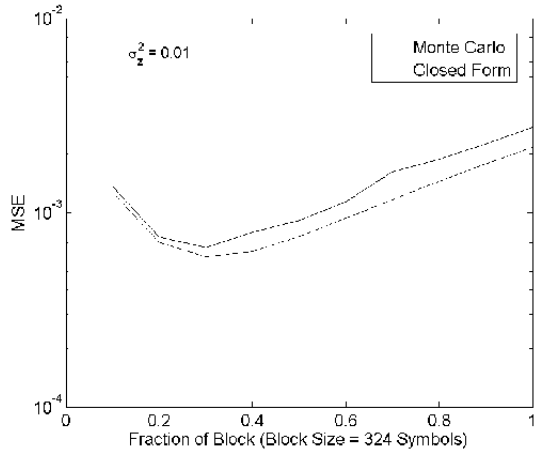


Figure 2: Behavior of MSE of a 4×4 channel for $f_D T_s = 7.72 \times 10^{-5}$ with block length of 324 symbols.

From Fig. 2 we see that for about 30 percent of a code block of length 324 symbols, lowest MSE is achieved for $f_D T_s = 7.72 \times 10^{-5}$. We can use this as a guideline to set the length of symbols in our tracking algorithm. The tracking algorithm is explained in Section 3.2.3.

Fig. 3 shows the MSE for our channel tracking algorithm. For a setup similar to above, the lowest MSE for channel estimation is achieved when only about 30 percent of code block is used. The next section gives a brief explanation of the D-BLAST architecture and then the channel tracking algorithm.

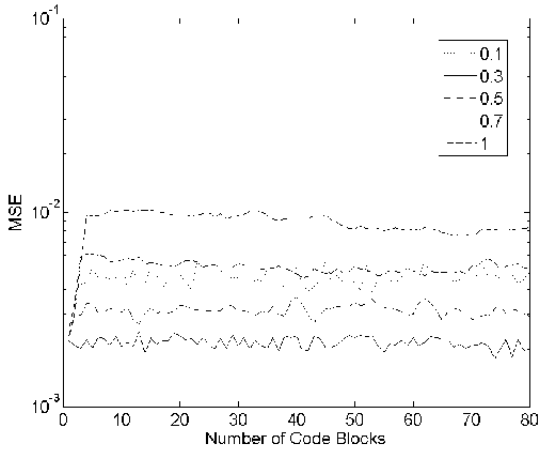


Figure 3: 4×4 D-BLAST tracking MSE w.r.t sub-blocks for $f_D T_s = 7.72 \times 10^{-5}$.

3.2.2 D-BLAST Architecture

D-BLAST stands for Diagonal Bell Labs Space Time Architecture. It is a transceiver architecture for MIMO systems [5]. For an $r \times t$ (r receive antennas and t transmit antennas)

MIMO system, the input data stream is demultiplexed into t streams. We call each demultiplexed stream as a sub stream. The t sub streams are then independently coded and modulated. These t sub streams are then transmitted over the t transmit antennas. However, the association between the sub streams and the antennas is changed in a periodic manner. Each sub stream is rotated through the t transmit antennas during one code block duration. This results in diagonal layering of the sub streams across the antennas and time. At the receiver end, decoding happens one layer at a time. Because of the diagonal structure, each sub stream sees channels from all the transmit antennas and hence has improved diversity compared with V-BLAST [6]. This results in spreading out the errors, in the event that one of the links has a bad channel condition, thereby minimizing the probability of outage. The diagonal structure renders D-BLAST as an outage optimal scheme in slow fading environments [2].

3.2.3 Channel Tracking

For the sake of illustration we describe our scheme for a 2×2 MIMO system. It can be extended to any $q \times q$ MIMO system in a straightforward manner. The symbol space-time diagram is shown below. The stream of symbols $\mathbf{x}_{ij}$ denote that they belong to stream i and are transmitted during the j^{th} time block. t_k and a_k denote the time block and antenna number respectively, over which the indexed symbols are transmitted. We use the j^{th} time block abstraction to indicate the time duration during which the transmitter keeps the association between the streams and the antennas constant. $\mathbf{p}$ indicates pilot symbols. l_i denotes i^{th} layer. For example: $x_{11}x_{12}$ is 1^{st} layer, $x_{21}x_{22}$ is 2^{nd} layer and so on.

	t_0	t_1	t_2	t_3	...
a_1	$\mathbf{p}_{a_1}$	$\mathbf{x}_{11}$	$\mathbf{x}_{21}$	$\mathbf{x}_{13}$	...
a_2	$\mathbf{p}_{a_2}$	$\mathbf{p}_{a_2}$	$\mathbf{x}_{12}$	$\mathbf{x}_{22}$	...

Table 1: Symbol Space-Time Diagram

Let $\mathbf{Y}_i$ and $\mathbf{X}_i$ be the received and transmitted set of vector in time block i. We assume that we already have the initial channel estimate $\hat{\mathbf{H}}_0$, found during time block $t_0$using the pilots $\mathbf{p}_{a_1}$ and $\mathbf{p}_{a_2}$ as shown in previous section. We also assume that the pilot $\mathbf{p}_{a_2}$ has been subtracted out from the received vector during t_i.

In this scheme, we use the initial estimate $\hat{\mathbf{H}}_0$ to detect $\mathbf{x}_{11}$ and $\mathbf{x}_{12}$. The receiver then decodes, re-encodes and performs baseband modulation to reconstruct $\hat{\mathbf{x}}_{11}$ and $\hat{\mathbf{x}}_{12}$. $\hat{\mathbf{x}}_{11}$along with $\mathbf{p}_{a_2}$ are used to find a new estimate $\hat{\mathbf{H}}_1$ of the channel. The tensor product of $\hat{\mathbf{x}}_{12}$ and its corresponding channel vector from $\hat{\mathbf{H}}_1$ is then subtracted from $\mathbf{Y}_2$, where $\mathbf{Y}_2$ is the channel output during the time block t_2. This new channel estimate $\hat{\mathbf{H}}_1$ is then used to detect the next layer i.e., $\mathbf{x}_{21}$ and $\mathbf{x}_{22}$, and the process is repeated.

3.3 Results and Discussion

3.3.1 Channel Tracking

For simulation purposes we consider a 4 x 4 MIMO system with a Rayleigh fading channel. For error control coding we use an LDPC (Low Density Parity Check) code of block length 1944. The bit streams are then modulated by 4-QAM modulators. We show simulation results for 2 cases of channel variations and 2 cases of code rates.

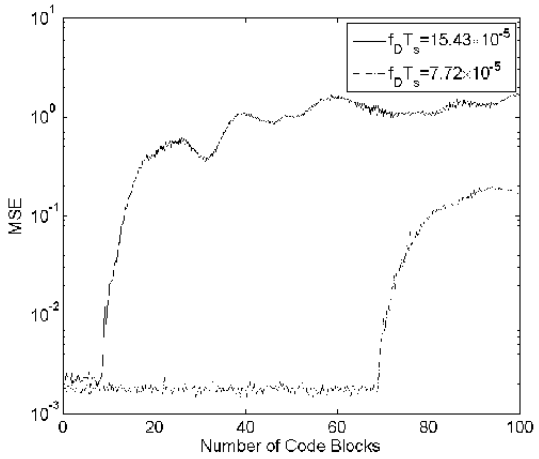


Figure 4: Behavior of channel estimation mean square error with increasing number of code blocks

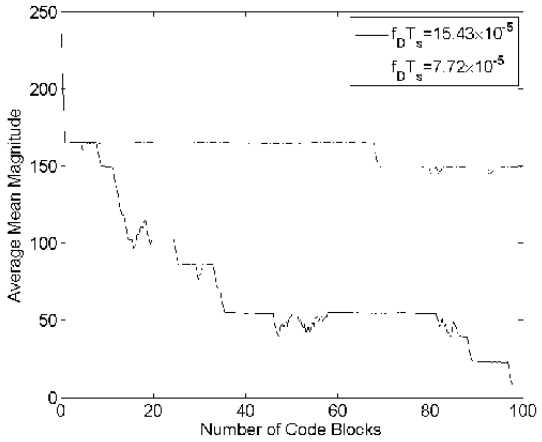


Figure 5: Behavior of the mean magnitude of LLR of decoded LDPC code with increasing number of code blocks

We can see that for a faster varying channel, the algorithm loses track very early. The sudden jumps in MSE in Figure 4 indicate loss of channel tracking. They correspond to a sudden change in the mean magnitude of LLRs of decoded LDPC codes in Figure 5. We can use these sudden changes to detect outage at the receiver.

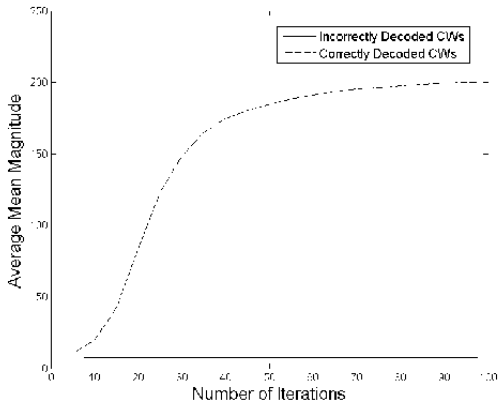


Figure 6: Behavior of Average Mean Magnitude of Log Likelihood Ratios for an LDPC code with $N = 648$ and $R = \frac{1}{2}$

3.3.2 Outage Detection

During the tracking process, decoding errors may occur. The incorrectly decoded codeword will lead to an inaccurate channel estimate. In order for the receiver to detect this, and thereby tell the transmitter to reinitialize the transmission, we use the mean magnitude of LLRs of LDPC codes [7]. We found through simulations (see Fig. 6) that the mean magnitude of LLRs of correctly decoded LDPC codes increases with the number of message passing iterations. In comparison, for incorrectly decoded codewords, the mean magnitude remains constant at some small value. We can use this property to detect the loss in tracking. Loss in tracking results in incorrectly decoded codewords or vice versa, thereby resulting in smaller mean magnitude of the LLRs. Using a threshold detector we can detect this loss in tracking and thus detect outage.

3.4 Channel Tracking Simulator

In the previous sections we studied channel tracking for a D-BLAST communication system using payload data. In this section we provide an overview of the tool we developed to simulate such channel tracking. Using this tool we can visualize the performance of the channel tracking algorithm for different set of parameters with ease. The tool was developed in MATLAB [8].

3.4.1 Component Description

There is only one major component to the GUI - the main window as shown in 7. It is used to input/select different parameters, which are then passed on the appropriate functions for simulation. We give a brief description of the components of the main window below.

- **# of Transmit Antennas** is use to set the number of transmit as well as receive antennas we assume in out calculations that they are equal in number. Inputting a non-integer value will cause an error.

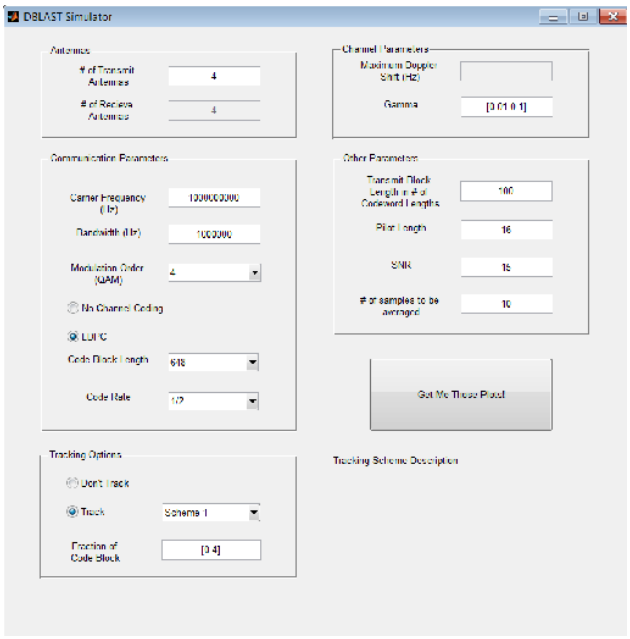


Figure 7: Channel Tracking Simulator

- **# of Receive Antennas** is not editable. Its value is set equal to that of Transmit Antennas.
- **Carrier Frequency** is used to set the carrier frequency in Hertz.
- **Bandwidth** is used to set the communication bandwidth in Hertz.
- **Modulation Order** is used to select either BPSK (2) or QPSK (4) mode of modulation.
- **No Channel Coding** sets the simulation to not to use channel coding. If this item is selected the following item *LDPC* cannot be selected.
- **LDPC** sets the simulation to use LDPC codes during simulation. There are in all 6 combinations while using LDPC codes. Code blocks of lengths 648, 1296 and 1944. While code rates options are 3/4 and 1/2. The parity check matrices for these codes are obtained from 802.11n standard [9].
- **Don't Track**, if selected doesn't perform any tracking. It is suitable only if we keep the channel constant.
- **Track** enables the selected scheme for channel estimation or tracking, depending on the scheme. The scheme descriptions are given below.
  - **Scheme 1** - the tracking scheme given in section 3.2.3.
  - **Scheme 2** - pilots multiplexed with data in the initial $t-1$ sub blocks.

  - **Scheme 3** - same as scheme 1, except that in this case we can set how much fraction of code-block to be used for channel tracking, whereas in scheme it was fixed to one sub block.
  - **Scheme 4** - a scheme, wherein we distribute the pilots across the payload.
  - **Fraction of Code Block** - this value has different interpretations. For Scheme 1 & 2 it is of no use. For Scheme 3, it means the fraction of code block to be used for channel estimation. In this case we can set it as an array of different values. For Scheme 4, it indicates the fraction of code block between two pilot symbol blocks. The value should not be an array in this case.

- **Maximum Doppler Shift**. This edit box is not functional.
- **Gamma** is used to set the maximum Doppler shift. Gamma is define as, $\gamma = \frac{\text{time to send one code block}}{\text{channel coherence time}}$. We use the gamma values to calculate maximum Doppler shift, which are then used to generate the channel coefficients.
- **Transmit Block Length in # of Codeword Lengths** is self-explanatory.
- **Pilot Length** is used to set the length of pilot block in number of symbol periods.
- **SNR** is used to set the average signal to noise ratio at each receiver antenna in decibels.
- **# of samples to be averaged** are the number of Monte Carlo runs that we average to get the sample statistics.

3.5 Conclusion

In this chapter we studied channel tracking for a D-BLAST communication system using payload data. Since the tracking involves using previously detected data symbols, a closed-form expression for channel MSE as a function of the number of training symbols, maximum Doppler spread and symbol duration was derived for a SISO link in a diffuse scattering environment. This expression was then extended to MIMO links. We use this analysis to find the optimum length of previously detected symbols for channel tracking. In the event the receiver loses track of the channel, we show, using simulations, that using the mean magnitude of log likelihood ratios of LDPC codes can detect the event of a decoding error, which in turn can be used to detect outage.

We have also provided a MATLAB based channel tracking simulator. It can be used to easily simulate the developed channel tracking and outage detection schemes.

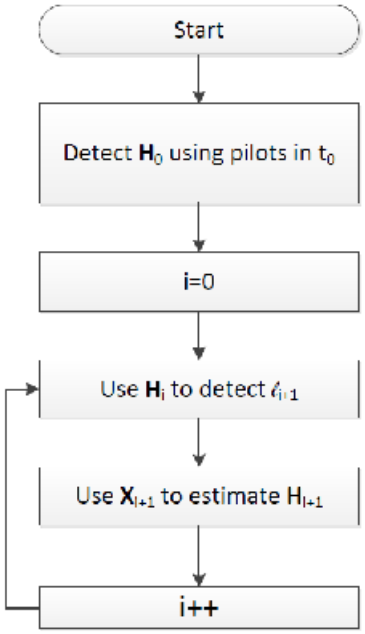


Figure 8: Scheme 1 Flowchart. $l_i = [\mathbf{s}_{1,i}\, \mathbf{s}_{2,i+1} \ldots\, \mathbf{s}_{t,i+t-1}]$, $\mathbf{H}_i = [\mathbf{h}_1^{(i)}\, \mathbf{h}_2^{(i)} \ldots\, \mathbf{h}_t^{(i)}]$

Variable Rate MIMO

4.1 Introduction

The use of autonomous/unmanned vehicles for various civilian and military applications has become increasingly prevalent. The absence of human pilots on board those unmanned systems, however, make it a challenging task to accomplish various intended missions. For example, these unmanned systems are often remotely piloted and thus having a reliable and robust communication link between the aerial systems and the ground control unit is imperative. Even for systems that are semi-autonomous, having a high throughput communication link is often essential to accomplishing any intended mission. Many remotely piloted aircrafts (RPAs) are used for surveillance applications and the need to stream surveillance data, including real time video data requires a highly reliable and high-throughput communication link from the RPAs to ground units.

Many legacy communication links (e.g., Link 16 with a data rate not more than 16kb/s) operate at a data rate that becomes highly inadequate for applications where video streaming from aerial systems to ground is needed. Merely scaling up the power/bandwidth is both limited by resource and policy constraints as well as the fundamental theoretical limits dictated by the Shannon theory. A promising technology is the use of multiple antenna communication systems [5, 10, 11]. The so-called multiple-input multiple-output (MIMO) communication scales up data rate linearly as the number of antennas increase and thus provides great potential for improving the throughput of air to ground communication. This helps enable many envisioned applications that may otherwise be infeasible.

MIMO communications, however, are traditionally designed for the so-called scattering environment [2] where independent channel variations between different transmit/receive antenna pairs are exploited. For airborne platforms, however, there has been a debate about the feasibility of MIMO communications because of the lack of scattering. However, for certain communication ranges, the large aperture that an aircraft affords makes MIMO an appealing choice of communication that can attain significantly higher throughput given a fixed power/bandwidth budget compared with single antenna systems even in the absence of any scatterers [12].

This chapter describes an ongoing research and development effort that uses MIMO communications to enable robust and high capacity connectivity between RPAs and ground

terminals. While the large aperture may compensate for the lack of the scattering, two unique challenges still exist for airborne MIMO communications. The large aperture is only attained when antennas are placed strategically apart on a RPA. In the absence of scattering, i.e., when communications are limited by line-of-sight channels, the fact that some antenna elements on a RPA may be completely out of sight from its communicating party may render the channel matrix ill-conditioned. This is further complicated by the high mobility and maneuverability of the RPA which make it infeasible to have complete channel state information (CSI) at the transmitter. To address these challenges, we propose a variable rate MIMO communication scheme that combines the D-BLAST architecture with per antenna spreading to harvest the maximum possible throughput gain allowed by the channel.

The chapter is organized as follows. Section 4.2.1 describes the channel measurement apparatus. The measurement data provides guidance on the potential throughput gain for the particular application of interest as well as challenges in realizing the throughput gain to address these challenges. Section 4.2.2 introduces a modified D-BLAST architecture to address these challenges. Section 4.3 describes simulation results using the real measurement channels to show the potential improvement using the proposed variable MIMO scheme.

We use the following notations throughout the chapter: * denotes complex conjugate, $\mathcal{CN}$ stands for complex Gaussian, ^{T} is transpose, $^+$ is conjugate transpose, $\mathbf{0}$ is a zero vector and $\mathbf{I}_k$ is a $k \times k$ identity matrix. Small letters denote scalars, bold small letters denote column vectors and bold capital letters denote matrices.

4.2 Methods, Assumptions and Procedures

4.2.1 MIMO Channel State Matrix Measurements

Measurement Apparatus

The process of measuring the channel state matrix at the Newport NY radio range required measuring the complex transmission coefficient from each antenna element on the ground array to each element on the unmanned combat air vehicle (UCAV) model of a RPA. The UCAV is shown in Fig. 9.

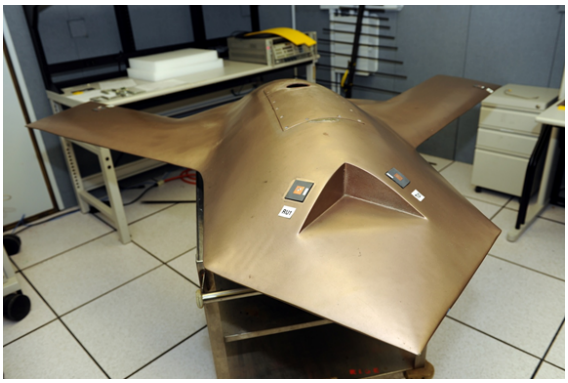
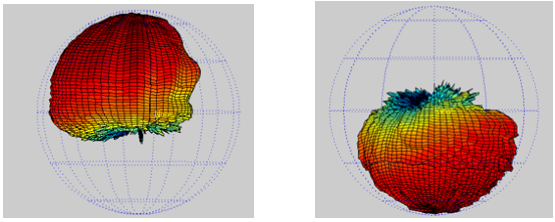


Figure 9: UCAV with square patch antennas.

The UCAV model was coated with Electrodag to shield the cable networks placed inside

the model that connect the patch antennas to the switch which feeds the RF (Radio Frequency) receiver. Although this shielding is not necessary in an operational RPA, it helps, in analyzing the measurements, to restrict the received signals to the patch antennas only. Four patches were installed for the uplink band (at carrier frequency 5.1 GHz with a bandwidth of 100 MHz) and four patches for the downlink band (at carrier frequency of 5.8GHz with a bandwidth of 100 MHz). Each patch has two ports which provide perpendicular linear polarization with about 30 dB cross polarization discrimination (isolation between ports). Thus 8 antenna ports are provided for the uplink and for the downlink, respectively. For easy reference these ports are uniquely labeled. For example, RU12 represents the second port on the first uplink patch on the upper side of the UCAV. The antenna patterns were measured on the Newport Range. Two such power patterns are shown in Fig. 10(a) and 10(b).



((a)) Antenna RU11 ((b)) Antenna RU31

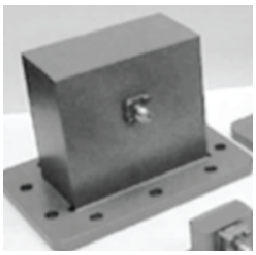
Figure 10: Antenna Pattern

The power pattern for antenna RU11 gives pretty good coverage above the UCAV because it is on the upper surface of the UCAV. Antenna RL31 is on the bottom surface of the UCAV wing and has good coverage in the hemisphere below the UCAV but is weak above the UCAV. These patterns emphasize the need for the D-BLAST (Diagonal Bell Labs Layered Space Time) [5] form of MIMO in order to provide full coverage for all data streams.

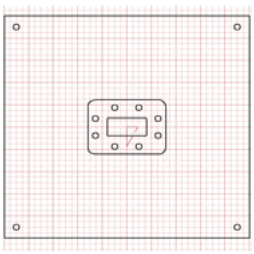
The ground array utilizes coax-to-waveguide adapters as antenna elements. They provide wide angular coverage (110^o E-Plane and 80^o H-Plane) and wide bandwidth 4.9 GHz to 7.05GHz. The array is mounted with square plates as shown in Fig. 11. The square plates allow a choice of vertical or horizontal polarizations. Typically measurements are made with all vertically polarized antenna elements or with polarizations alternated between vertical and horizontal polarization to see the effect of polarization mixing on MIMO capacity. The elements can be at various spacing by bolting the plates to different positions along the mounting rails. The spacing shown in Fig. 11 provide an 80 inch wide array. Another test used a 30 foot wide array. The array is mounted on an expanding tower which allows array heights of 2 feet to 40 feet above ground level.

The UCAV is mounted on a positioning tower at a height of 70 feet. The position of the UCAV can be rotated around a vertical axis (relative to ground) for 360 degrees of azimuth. It can also be rotated in pitch and roll over 90. The UCAV support is provided by an absorber covered arm from the top or bottom of the UCAV. A view of the 400 foot range at Newport showing the UCAV and the ground array is shown in Fig. 12. It is seen that the range is relatively free of scattering, so that the MIMO performance should be similar to free space limitations.

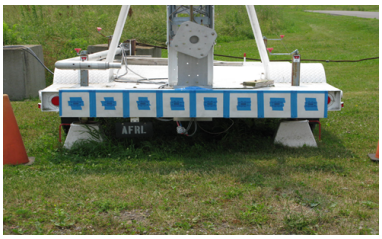
The switch at the ground array cycles through the 8 elements in a 20 millisecond period. After each of the ground array cycles, the switch at the UCAV connects to a new antenna



((a)) Plate



((b)) Plate Schematic



((c)) Ground Array

Figure 11: Element Ground Array

Figure 12: UCAV and Ground Array on the 400 ft. Newport Range.

port of the 16 UCAV antenna ports. As this switching is completed the azimuth turntable moves to a new position for a given pattern cut. At each position of the cut, the complex transmission coefficient from each antenna element on the ground array to each element port on the UCAV is recorded. The frequency and pitch and yaw positions can be modified for new cuts.

Challenges

Fig. 10(a) and 10(b) illustrate the unique challenges of airborne MIMO. Each antenna element has only limited visibility for communication, depending on the orientations of the transceiver pair. This is further exacerbated by the lack of channel state information at the transmitter - RPAs are not only of high mobility but they may maneuver in-flight which makes it infeasible to constantly feedback channel state to the transmitter.

Nevertheless, the channel matrices indicate that there exists significant theoretical through-

put improvement through MIMO communications if designed properly. In the following, we describe a simple variable rate MIMO scheme to address the above mentioned challenges. We demonstrate through numerical simulation that the proposed scheme is capable of overcoming the above challenges and realizing significant performance gain.

Measurement Data

We obtain measurement data for the following two configurations.

LRPRM This configuration consists of setting the 80 inch wide ground array at 2 feet above the ground level. The polarization of the antenna elements is alternated between vertical and horizontal polarization. The UCAV has the same configuration as mentioned in the previous section.

LRPRV Same as LRPRM except all the antenna elements transmit with vertical polarization. This configuration consists of setting the 80 inch wide ground array at 2 feet above the ground level, but all the antenna elements transmit with vertical polarization.

4.2.2 Variable Rate MIMO

While the channel matrix measurements exhibit potential for significant throughput improvement via MIMO communications, significant challenges exist that need to be overcome in order to realize this potential. In particular, the lack of scattering makes the channel susceptible to ill-conditioning when antenna elements may be out of sight to the communicating party. In the absence of complete CSI at the transmitter, there is a need to ensure that any independent data stream is not stuck with an antenna element that is out-of-sight. This makes D-BLAST a very natural candidate for such applications. D-BLAST is an outage optimal transceiver architecture for MIMO communication system [2, 5]. The architecture essentially involves independent coding and modulation of N data streams and rotating each stream through all the transmit antennas. Here, N is the number of transmit antennas. The rotation is performed in a way that makes sure that each stream experiences the channel from all the transmit antennas and all levels of interference *i.e.,* no interference to experiencing interference from all of the other $N-1$ streams at the receiver as each stream rotates in the space-time domain.

The verbatim application of D-BLAST, however, is still inadequate when channel matrices are highly ill-conditioned. The existence of antennas that experience channel outage due to blockage will drag down the overall performance. From a theoretical viewpoint, the independent data streams accommodated in a MIMO channel should be no larger than the effective rank of the matrix. The challenge is, in the absence of complete CSI at the transmitter, how to utilize the advantage of D-BLAST while being able to handle the potential rank deficiency of channel matrices.

We propose a simple scheme of variable rate MIMO - the variable rate is not only achieved though the traditional means of controlling the coding rate as well as modulation order, but the number of independent data streams is also adapted through per antenna spreading. The modified D-BLAST architecture is illustrated in Fig. 5. Without loss of generality, we assume the MIMO system has N transmit and N receive antennas. At the transmitter side,

spreading is done on a per antenna basis with a length M Walsh code for each antenna where $M < N$. As such, a group of N/M antennas will share the same Walsh code and the effective number of data streams is reduced to N/M. At the receiver side, de-spreading is embedded in the D-BLAST receiver to facilitate successive interference cancellation.

As an illustration, consider an 8×8 MIMO system. Let $a_0, a_1, \ldots, a_7$ denote the 8 transmit antennas. Let each symbol from any 4 antennas, say a_0, a_1, a_2, a_3 be spread by $[1\ 1]^T$. Similarly, let each symbols from the remaining 4 antennas, a_4, a_5, a_6, a_7 be spread using $[-1\ 1]^T$. Notice that $[1\ 1]^T$ and $[-1\ 1]^T$ are orthogonal to each other. At the receiver, de-spreading effectively reduces the number of mutually interfering streams from 8 to 4.

The illustration in the previous paragraph implicitly assumes that the transmitter and the receiver have agreed on some spreading factor, M, and the corresponding Walsh codes. As the transmitter does not possess any CSI, our assumption holds true only if there exists a very low rate feedback link from the receiver to the transmitter. The receiver, which knows the CSI, can feedback the spreading factor to the transmitter, which then can adapt the transmission strategy to the channel conditions. For example, in the 8×8 case, one needs a total of 2 bits for the spreading factor of 1 (no spreading), 2, 4, and 8 where the last one corresponds to essentially a beamforming strategy.

In the next section we simulate the variable rate MIMO scheme over measured channels to examine the performance of the proposed variable rate MIMO D-BLAST for low rank channels.

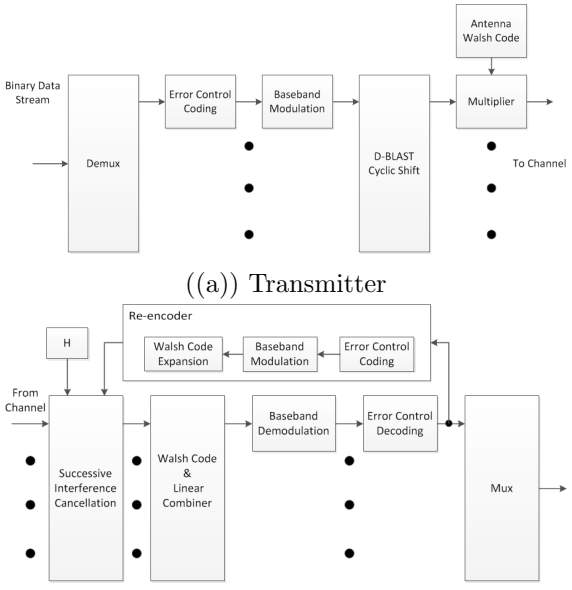


Figure 13: Variable rate MIMO transceiver block diagram

4.3 Results and Discussion

We assess the performance of the variable rate D-BLAST scheme by comparing it with the original D-BLAST for the following two channel measurement configurations.

- All Vertical Polarization (VP).
- Mixed Polarization (MP).

As mentioned earlier, we are motivated to use VP and MP to see the effect of polarization mixing on MIMO capacity. The channels obtained using VP tend be rank deficient. On the other hand, channels obtained using MP tend to have a higher rank than that of VP. This is because the two polarizations effectively serve to create two orthogonal channels. This leads to more well-conditioned MP channel matrices than the corresponding VP ones.

For each of the above two configurations we have obtained about 13,000 channel measurements, each of which corresponds to a unique combination of azimuth and elevation. During simulation, we select 10% of the channels at random, and measure the average outage over these channels.

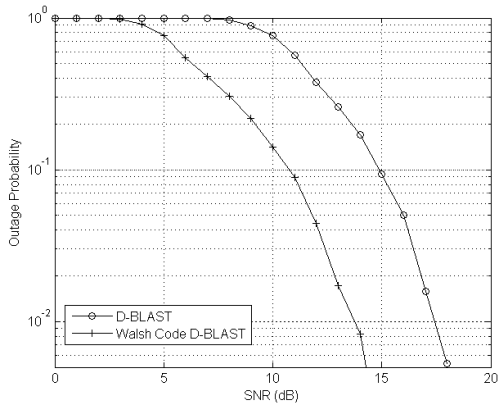


Figure 14: Outage rate for all vertical polarization at rate 4 bits/s/Hz.

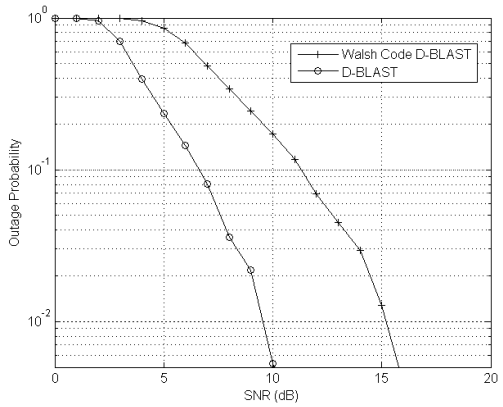


Figure 15: Outage rate for mixed polarization at rate 4 bits/s/Hz.

Fig. 14 compares the outage behavior of the variable rate D-BLAST scheme with the original D-BLAST at a fixed rate of 4 bits/s/Hz. The variable rate D-BLAST scheme uses a spreading factor of 2, QPSK modulation and 1/2 rate LDPC code. While the D-BLAST

scheme uses BPSK modulation and 1/2 rate LDPC code. Clearly, at low outage probabilities, the variable rate D-BLAST scheme provides an SNR gain of about 4 dB over the original D-BLAST.

Fig. 15, on the other hand, shows that the variable rate D-BLAST scheme is inefficient for mixed polarization channels. The original D-BLAST outperforms variable rate D-BLAST by about 6 dB SNR at low outage probabilities.

The above simulation results show that for the low rank channel matrices obtained by only vertical polarization, the variable rate D-BLAST scheme outperforms D-BLAST, while for well conditioned channel matrices, the original D-BLAST architecture suffices.

4.4 Conclusion

In this chapter we have introduced a new scheme to transmit data over MIMO channels which may be ill-conditioned. In the absence of channel state information at the transmitter the scheme overcomes this challenge by per antenna spreading in the D-BLAST architecture, thereby effectively reducing the number of independent data streams. Using real measurement channels we demonstrated through simulations that one can achieve significant SNR gain over the original D-BLAST for rank deficient channel matrices.

GnuRadio MIMO Implementation Using USRP N210

5.1 Introduction

To facilitate experimental validation of variable rate MIMO we have developed a software defined 2×2 MIMO system using GNU Radio/USRP platform [1,13]. In this setup the GNU Radio based software performs all the baseband signal processing, whereas USRPs act as the wireless interfaces. To form a 2 antenna terminal, two USRP N210 are chained together using a vendor supplied cable. Thereby, a total of 4 USRPs combined with the GNU Radio based baseband signal processing constitute our 2×2 MIMO implementation. Nevertheless, the software implementation is scalable to higher antenna dimensions under the constraint that the number of transmit antennas is equal to that of receive antennas and some changes to the symbol timing estimator in the receive chain.

An important feature of this implementation is the decoupling of GNU Radio with the DBLAST encoding/decoding functionality: The DBLAST decoder/encoder c++ library is developed separately which is then used as a shared library by the GNU Radio modules. Such separation increases the potential of code re-usability. We will revisit this aspect of our implementation in Section 5.2.2. In the next section we provide high level block diagrams that represent the implementation approach.

5.2 Methods, Assumptions and Procedures

5.2.1 System Overview

Fig. 16 shows the transmitter block diagram. As shown, an input byte stream is first demultiplexed and then given to the *DBLAST Encoder*. The DBLAST encoder performs channel encoding, digital constellation mapping and the diagonal layering of the two incoming streams. The two output streams of the encoder are then fed to two upsampling *RRC Filter*s (root raised cosine). The outputs of these filters are then fed to two separate USRP N210s, which share a common 10 MHz reference clock.

Fig. 17 shows the structure of the receiver. The outputs from the two synchronized

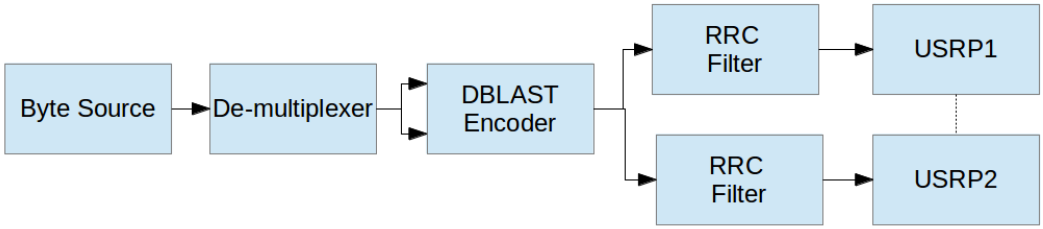


Figure 16: Transmitter Block Diagram

USRP N210s are RRC filtered and processed to obtain the symbol timing and then sampled accordingly. The sampled output is then compensated for the carrier frequency offset (CFO) between the transmit and receive USRPs. The CFO corrected output is then given to the *DBLAST Decoder* which performs MMSE SIC detection on the received signal.

5.2.2 C++ MIMO Library

As mentioned previously, we have implemented DBLAST encoding / decoding as a separate C++ library which is then used by our GNU Radio implementation. The library can be used independently of GNU Radio and makes use of Eigen library [14] for matrix manipulations. The DBLAST decoder therein implements MMSE SIC algorithm. The library along with its documentation has been delivered to AFRL.

5.3 Results and Discussion

Using the GNU Radio/USRP software radio platform we successfuly implemented a DBLAST transciever. We also performed over the air tests with success and thereby validated the implementation. An independent c++ library that provides DBLAST encoder/decoder functionality was developed and delivered as well.

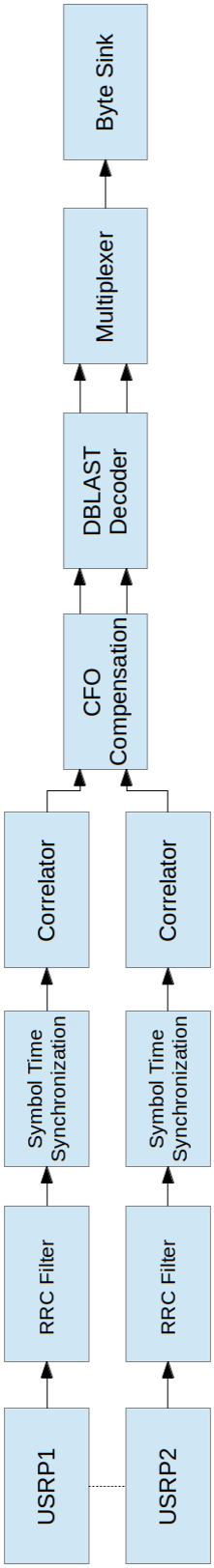


Figure 17: Receiver Block Diagram

Conclusion

In this report we studied problems relating to airborne MIMO communication and delivered relevant MATLAB, GNU Radio and C++ based simulation and experimentation software.

Firstly, we studied the problem of channel tracking in dynamic conditions entailed in airborne MIMO communication. In doing so, we developed a scheme to estimate and keep the channel state information current using payload data. We analyzed the scheme and provided a closed form expression to determine optimal symbol length needed to estimate the channel in a continuously fading Rayleigh environment. In addition, we have provided a channel tracking simulator, developed in MATLAB, to quickly test different scenarios.

Secondly, by studying the real measurement data, we found out that the wireless channel matrices offered therein are severely rank deficient. Traditional MIMO architectures, which are developed to take advantage of well conditioned MIMO channel matrices, fail to take advantage of such rank deficient channels. We developed a Variable Rate MIMO scheme to overcome the problem of communication over rank deficient channels. Through simulations we showed that the developed scheme significantly outperforms D-BLAST for rank deficient channel matrices.

Finally, we provided a GNU Radio based 2×2 D-BLAST implementation, to facilitate experimentation. Further, we developed a C++ library that implements D-BLAST encoder and decoder.

References

[1] USRP: Universal Software Radio Peripheral, accessed September 2013. [Online]. Available: http://www.ettus.com

[2] D. Tse and P. Viswanath, *Fundamentals of Wireless Communications*. Cambridge, UK: Cambridge University Press, 2005.

[3] Q. Sun, D. Cox, H. Huang, and A. Lozano, "Estimation of continuous flat fading MIMO channels," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 549–553, 2002.

[4] V. Pohl, P. Nguyen, V. Jungnickel, and C. von Helmolt, "Continuous flat-fading MIMO channels: Achievable rate and optimal length of the training and data phases," *IEEE Trans. Wireless Commun.*, vol. 4, no. 4, pp. 1889–1900, 2005.

[5] G. J. Foschini, "Layered Space-Time Architecture for Wireless Communication in a Fading Environment When Using Multi-Element Antennas," *Bell Laboratories Technical Journal*, vol. 1, no. 2, pp. 41–59, 1996.

[6] G. Foschini, D. Chizhik, M. Gans, C. Papadias, and R. Valenzuela, "Analysis and performance of some basic space-time architectures," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 303–320, 2003.

[7] J. Li, X. You, and J. Li, "Early stopping for LDPC decoding: convergence of mean magnitude (cmm)," *IEEE Commun. Lett.*, vol. 10, no. 9, pp. 667–669, 2006.

[8] MATLAB, *7.13.0.564 (R2011b)*. Natick, Massachusetts: The MathWorks Inc., 2011.

[9] IEEE 802.11n-2009 (Section 20.3). http://standards.ieee.org/ndstds/standard/802.11n-2009.html.

[10] G. J. Foschini and M. J. Gans, "On Limits Of Wireless Communications in a Fading Environment When Using Multiple Antennas," *Wireless Personal Commun.*, vol. 6, no. 3, pp. 311–335, 1998.

[11] E. Telatar, "Capacity of Multi-antenna Gaussian Channels," *Europ. Trans. Telecommun.*, vol. 10, no. 6, pp. 585–595, 1999.

[12] M. J. Gans, "Aircraft Free-Space MIMO Communications," in *Proc 43rd Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, Nov. 2009.

[13] GNU Radio, accessed September 2013. [Online]. Available: http://www.gnuradio.org

[14] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.

Symbols, Abbreviations and Acronyms

$\mathbf{0}$ zero vector with appropriate dimension depending on the context

$\mathbf{I}_k$ identity matrix of dimension with k columns and k rows.

$\mathbf{X}$ matrix denoted by uppercase bold letter

$\mathbf{x}$ column vector denoted by lowercase bold letter

* complex conjugate operator

$^+$ Hermitian matrix operator

^{T} transpose operator

$r \times t$ wireless communication system with r receive antennas and t transmit antennas

x scalar denoted by lower case letter

BPSK binary phase shift keying

CSI channel state information

D-BLAST diagonal Bell laboratories layered space-time

GUI graphical user interface

LDPC low density parity check

MIMO multiple input multiple output

MSE mean square error

QAM quadrature amplitude modulation

QPSK quadrature phase shift keying

RF radio frequency

RPA remotely piloted aircraft

SNR signal to noise power ratio

UCAV unmanned combat air vehicle

Appendices

Appendix A

Proof of Theorem 1

Proof. Using the assumptions mentioned in the text, we first evaluate $\mathscr{E}(\Delta_p\Delta_p^*)$.

$$\begin{aligned}
&\mathscr{E}(\Delta_p\Delta_q^*)\\
&= \mathscr{E}\Bigg[\sum_{k=1}^{S}\beta_k\left(e^{j2\pi f_k(N-p)T_s}-e^{j2\pi f_kNT_s}\right)\\
&\quad\sum_{l=1}^{S}\beta_l^*\left(e^{-j2\pi f_l(N-q)T_s}-e^{-j2\pi f_lNT_s}\right)\Bigg]\\
&= \mathscr{E}\Bigg[\sum_{k=1}^{S}|\beta_k|^2\left(e^{j2\pi f_k(N-p)T_s}-e^{j2\pi f_kNT_s}\right)\\
&\qquad\left(e^{-j2\pi f_k(N-q)T_s}-e^{-j2\pi f_kNT_s}\right)\Bigg]\\
&\quad+\mathscr{E}\Bigg[\sum_{k\neq l}\beta_k\beta_l\left(e^{-j2\pi f_l(N-p)T_s}-e^{-j2\pi f_lNT_s}\right)\\
&\qquad\left(e^{-j2\pi f_l(N-q)T_s}-e^{-j2\pi f_lNT_s}\right)\Bigg]\\
&= \mathscr{E}\Big[\left(1-e^{-j2\pi f_kpT_s}\right)\left(1-e^{j2\pi f_kqT_s}\right)\Big]\,\mathscr{E}\left[\sum_{k=1}^{S}|\beta_k|^2\right]\\
&= \mathscr{E}\Big[\left(1-e^{-j2\pi f_kpT_s}\right)\left(1-e^{j2\pi f_kqT_s}\right)\Big]\\
&= 1-\mathrm{sinc}(2\pi f_DpT_s)-\mathrm{sinc}(2\pi f_DqT_s)\\
&\quad+\mathrm{sinc}(2\pi f_D(p-q)T_s),
\end{aligned}\tag{17}$$

where $\mathrm{sinc}(x)=\frac{\sin(x)}{x}$. Third equality is because the second expectation in the second step is zero.

From the above expression we see that,

$$\mathscr{E}(\Delta_p\Delta_q^*)=\mathscr{E}(\Delta_q\Delta_p^*).\tag{18}$$

Also, putting $p=q$ in expression (17), we get,

$$\mathscr{E}(\Delta_p\Delta_p^*)=2(1-\mathrm{sinc}(2\pi f_DpT_s)).\tag{19}$$

We now simplify the expectation term in equation (6). Using brute force we can show that,

$$\mathscr{E}\left[\left(\sum_{k=1}^{2}\Delta_k\right)\left(\sum_{k=1}^{2}\Delta_k^*\right)\right]$$
$$= 2\left[\sum_{k=1}^{2}k - \sum_{k=1}^{2}k\,\mathrm{sinc}(2\pi f_D k T_s)\right], \tag{20}$$
$$\mathscr{E}\left[\left(\sum_{k=1}^{3}\Delta_k\right)\left(\sum_{k=1}^{3}\Delta_k^*\right)\right]$$
$$= 2\left[\sum_{k=1}^{3}k - \sum_{k=1}^{3}k\,\mathrm{sinc}(2\pi f_D k T_s)\right]. \tag{21}$$

We now simplify the expectation term in equation (6). We use proof by induction. Suppose the following is true,

$$\mathscr{E}\left[\left(\sum_{k=1}^{N}\Delta_k\right)\left(\sum_{k=1}^{N}\Delta_k^*\right)\right]$$
$$= 2\left[\sum_{k=1}^{N}k - \sum_{k=1}^{N}k\,\mathrm{sinc}(2\pi f_D k T_s)\right]. \tag{22}$$

Now,

$$\mathscr{E}\left[\left(\sum_{k=1}^{N+1}\Delta_k\right)\left(\sum_{k=1}^{N+1}\Delta_k^*\right)\right]$$
$$= \mathscr{E}\left[\left(\sum_{k=1}^{N}\Delta_k + \Delta_{N+1}\right)\left(\sum_{k=1}^{N}\Delta_k^* + \Delta_{N+1}^*\right)\right]. \tag{23}$$

Let, $\left(\sum_{k=1}^{N}\Delta_k\right) = D$.

$$\mathscr{E}\left[\left(\sum_{k=1}^{N+1}\Delta_k\right)\left(\sum_{k=1}^{N+1}\Delta_k^*\right)\right]$$
$$= \mathscr{E}\left[(D+\Delta_{N+1})\left(D^*+\Delta_{N+1}^*\right)\right] \tag{24}$$
$$= \mathscr{E}(DD^*) + \mathscr{E}(D\Delta_{N+1}^*) + \mathscr{E}(\Delta_{N+1}D^*)$$
$$+\mathscr{E}(\Delta_{N+1}\Delta_{N+1}^*). \tag{25}$$

We know $\mathscr{E}(DD^*)$ from equation (22). We need to find the last three terms.

$$\mathscr{E}(D\Delta_{N+1}^*)$$
$$= \mathscr{E}\left[(\Delta_1+\Delta_2+\ldots+\Delta_N)\Delta_{N+1}^*)\right] \tag{26}$$
$$= \mathscr{E}(\Delta_1\Delta_{N+1}^*) + \mathscr{E}(\Delta_2\Delta_{N+1}^*) + \ldots + \mathscr{E}(\Delta_1\Delta_{N+1}^*) \tag{27}$$
$$= N - \sum_{k=1}^{N}\mathrm{sinc}(2\pi f_D k T_s) - \sum_{k=1}^{N}\mathrm{sinc}(2\pi f_D (N+1) T_s)$$
$$+\sum_{k=1}^{N}\mathrm{sinc}(2\pi f_D (N+1-k) T_s) \tag{28}$$
$$= N - N\,\mathrm{sinc}(2\pi f_D (N+1) T_s). \tag{29}$$

From equation (19),

$$\mathscr{E}(\Delta_{N+1}\Delta_{N+1}^*) = 2(1 - \text{sinc}(2\pi f_D(N+1)T_s)). \tag{30}$$

From equations (18) and (27),

$$\mathscr{E}(D\Delta_{N+1}^*) = \mathscr{E}(\Delta_{N+1}D^*). \tag{31}$$

From equations (22), (25), (29), (30) and (31), we get

$$\begin{aligned}
&\mathscr{E}\left[\left(\sum_{k=1}^{N+1}\Delta_k\right)\left(\sum_{k=1}^{N+1}\Delta_k^*\right)\right] \\
&= 2\left[\sum_{k=1}^{N}k - \sum_{k=1}^{N}k\,\text{sinc}(2\pi f_D k T_s)\right] \\
&\quad +2(N - N\text{sinc}(2\pi f_D(N+1)T_s)) \\
&\quad +2(1 - \text{sinc}(2\pi f_D(N+1)T_s)) \qquad (32) \\
&= 2\left[\sum_{k=1}^{N+1}k - \sum_{k=1}^{N+1}k\,\text{sinc}(2\pi f_D k T_s)\right]. \qquad (33)
\end{aligned}$$

The last term in the above expression is of the form $\sum_{k=1}^{N} k\text{sinc}(ck)$, where c is a constant.

$$\begin{aligned}
&\sum_{k=1}^{N}k\,\text{sinc(ck)} \\
&= \sum_{k=1}^{N}k\,\frac{\sin ck}{ck} \\
&= \frac{1}{c}\sum_{k=1}^{N}\sin ck \\
&= \frac{1}{c}\,\sin\left(\frac{Nc}{2}\right)\sin\left(\frac{(N+1)c}{2}\right)\csc\left(\frac{c}{2}\right). \qquad (34)
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mathscr{E}\left[\left(\sum_{k=1}^{N}\Delta_k\right)\left(\sum_{k=1}^{N}\Delta_k^*\right)\right] \\
&= 2\left[\sum_{k=1}^{N}k - \sum_{k=1}^{N}k\;sinc(2\pi f_D k T_s)\right] \qquad (35) \\
&= N(N+1) - \frac{\sin(\pi f_D N T_s)\sin(\pi f_D(N+1)T_s)}{\pi f_D T_s \sin(\pi f_D T_s)}. \qquad (36)
\end{aligned}$$

□

Appendix B

C++ Library Reference

MIMO Library

by

Kapil M. Borle
kmborle@syr.edu

Generated by Doxygen 1.7.6.1

Wednesday 24th September, 2014

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

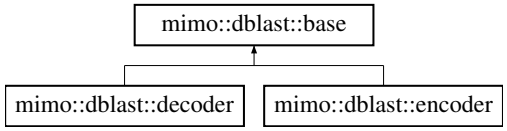
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 mimo::dblast::base Class Reference

Inheritance diagram for mimo::dblast::base:



Public Member Functions

- unsigned **ntx** () const
- unsigned **nrx** () const
- unsigned **message_length_bits** ()
- unsigned **block_length_bits** ()
- unsigned **block_length_symbs** ()
- unsigned **bits_per_symbol** ()
- unsigned **sub_block_length_symb** () const
- unsigned **num_sub_blocks** (const unsigned ncws)

base - Base Class for DBLAST encoder and decoder

Parameters

ntx	*- Number of transmit antennas*
nrx	*- Number of Receive antennas*
mod	*- Modulator object*
enc	*- Encoder object*

Returns

base object

- **base** (const unsigned ntx, const unsigned nrx, mimo::modulator &mod, mimo::encoder &enc)

Protected Member Functions

- void **set_num_sub_block** ()

Protected Attributes

- const unsigned **d_ntx**
- const unsigned **d_nrx**
- unsigned **d_sbl**
- unsigned **d_bls**
- mimo::repacker * **d_packer**
- mimo::modulator & **d_mod**
- mimo::encoder & **d_enc**

~base - default destructor

Returns

void

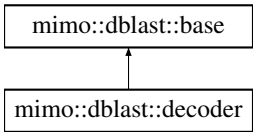
- ~**base** ()
- void **check_args** ()
- void **set_sbl** ()

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/dblast.h
- /home/kapil/libmimo/lib/dblast.cc

3.2 mimo::dblast::decoder Class Reference

Inheritance diagram for mimo::dblast::decoder:



Public Member Functions

decoder - DBLAST Decoder

Parameters

ntx	*- Number of transmit antennas*
nrx	*- Number of receive antennas*
mod	*- Modulator object*
demod	*- Demodulator object*
enc	*- Encoder Object*
denc	*- Decoder Object*
ninput_cws	*- Number of input codewords in a block*

Returns

decoder object

- **decoder** (const unsigned ntx, const unsigned nrx, mimo::modulator &mod, mimo::demodulator &demod, mimo::encoder &enc, mimo::decoder &denc, const unsigned ninput_cws)

set_ninput_cws - Allocates memory for use during encoding.

Parameters

ninput_cws	*- Number of input codewords*

Returns

void

- void **set_ninput_cws** (const unsigned ncws)

decode - Decodes received DBLAST signal

Parameters

inx	*- Matrix of dimenstion nsymb x ntx where nsymb is the number of symbols time slots and ntx is the number of transmit antennas*
Hhat	*- ntx nrx channel matrix*
snr	*- Signal to noise ratio.*

Returns

void

- void **decode** (const Eigen::MatrixXcf &inx, const Eigen::MatrixXcf &Hhat, const float &snr)

output_mat - Get decoded bits

Returns

Returns constant reference to Matrix of type mimo::MatrixXucharb. Each column corresponds to a decoded codeword.

- const MatrixXuchar & **output_mat** () const

output - Returns the output generated by the decode function

Parameters

out	*- Output vector to filled with decoded bits*

Returns

void

- void **output** (vector< uchar > &out) const

∼decoder - default destructor

Returns

void

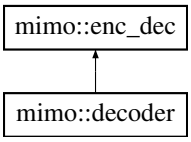
- ∼**decoder** ()

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/dblast.h
- /home/kapil/libmimo/lib/dblast.cc

3.3 mimo::decoder Class Reference

Inheritance diagram for mimo::decoder:



Public Member Functions

encoder - Generic decoder class

Parameters

n	*- Codeword block length in bits*
k	*- Codeword message length in bits*

Returns

decoder object

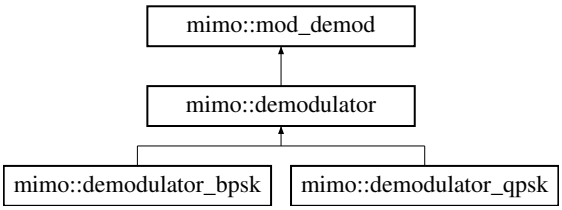
- **decoder** (unsigned n, unsigned k)
- ∼**decoder** ()
- virtual void **decode** (const vector< uchar > &x, vector< uchar > &y)
- virtual void **decode** (const uchar *inp, const unsigned inlen, uchar *outp, const unsigned outlen)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/enc_dec.h
- /home/kapil/libmimo/lib/enc_dec.cc

3.4 mimo::demodulator Class Reference

Inheritance diagram for mimo::demodulator:



Public Member Functions

∼**modulator_qpsk - default destructor**

Returns

void

- **demodulator** ()
- **demodulator** (const vector< complex< float > > &constellation)
- ∼**demodulator** ()
- virtual unsigned **decode** (const complex< float > &sample)
- virtual void **decode** (const complex< float > *sample, uchar *out, const unsigned vlen)
- virtual void **decode** (const complex< float > *sample, unsigned *out, const unsigned vlen)
- virtual void **decode** (const vector< complex< float > > &samples, vector< unsigned > &out)

Protected Attributes

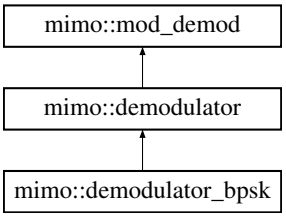
- vector< float > **d_distances**
- unsigned(demodulator::* **decode_sample_ptr**)(const complex< float > &)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.5 mimo::demodulator_bpsk Class Reference

Inheritance diagram for mimo::demodulator_bpsk:



Public Member Functions

~modulator_qpsk - default destructor

Returns

void

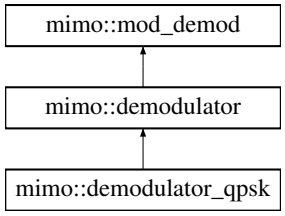
- **demodulator_bpsk** ()
- ~**demodulator_bpsk** ()
- unsigned **decode** (const complex< float > &sample)
- void **decode** (const complex< float > *sample, uchar *out, const unsigned vlen)
- void **decode** (const complex< float > *sample, unsigned *out, const unsigned vlen)
- void **decode** (const vector< complex< float > > &samples, vector< unsigned > &out)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.6 mimo::demodulator_qpsk Class Reference

Inheritance diagram for mimo::demodulator_qpsk:



Public Member Functions

∼modulator_qpsk - default destructor

Returns

void

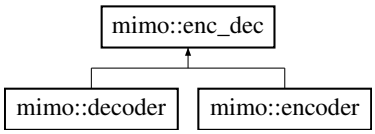
- **demodulator_qpsk** ()
- ∼**demodulator_qpsk** ()
- unsigned **decode** (const complex< float > &sample)
- void **decode** (const complex< float > ∗sample, uchar ∗out, const unsigned vlen)
- void **decode** (const complex< float > ∗sample, unsigned ∗out, const unsigned vlen)
- void **decode** (const vector< complex< float > > &samples, vector< unsigned > &out)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.7 mimo::enc_dec Class Reference

Inheritance diagram for mimo::enc_dec:



Public Member Functions

- unsigned **block_length_bits** ()
- unsigned **message_length_bits** ()

enc_dec - Base class for encoder and decoder class

Parameters

n	- *Codeword block length in bits*
k	- *Codeword message length in bits*

Returns

enc_dec object

- **enc_dec** (unsigned n, unsigned k)
- ~**enc_dec** ()

Protected Member Functions

- string **int_to_string** (int x)

Protected Attributes

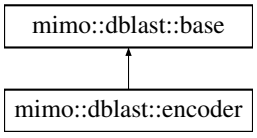
- unsigned **d_n**
- unsigned **d_k**

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/enc_dec.h
- /home/kapil/libmimo/lib/enc_dec.cc

3.8 mimo::dblast::encoder Class Reference

Inheritance diagram for mimo::dblast::encoder:



Public Member Functions

encoder - DBLAST Encoder Constructor

Parameters

ntx	*- Number of transmit antennas*
nrx	*- Number of receive antennas*
mod	*- Reference to modulator object*
enc	*- Reference to encoder object*
ninput_cws	*- Number of input codewords in a block*

Returns

DBLAST Encoder Object

- **encoder** (const unsigned ntx, const unsigned nrx, mimo::modulator &mod, mimo::encoder &enc, const unsigned ninput_cws)

∼encoder - default destructor

Returns

void

- ∼**encoder** ()

set_ninput_cws - Allocates memory for use during encoding.

Parameters

ninput_cws	*- Number of input codewords*

Returns

void

- void **set_ninput_cws** (const unsigned ninput_cws)

encode - Encodes the input stream of bits for DBLAST transmission

Parameters

inx	*- Input vector of type mimo::uchar. Vector length must be multiple of the codeword block length in bits.*

Returns

void (To access the result, call output function.)

- void **encode** (const vector< uchar > &inx)

output

Returns

returns reference to the output matrix of type Eigen::MatrixXcf

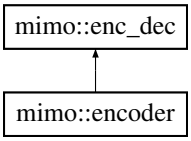
- const Eigen::MatrixXcf & **output** ()

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/dblast.h
- /home/kapil/libmimo/lib/dblast.cc

3.9 mimo::encoder Class Reference

Inheritance diagram for mimo::encoder:



Public Member Functions

encoder - Generic encoder class

Parameters

n	- *Codeword block length in bits*
k	- *Codeword message length in bits*

Returns

encoder object

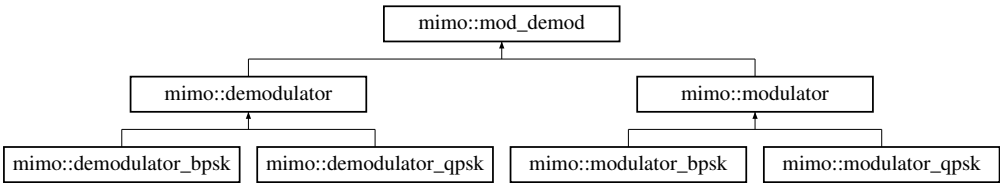
- **encoder** (unsigned n, unsigned k)
- ∼**encoder** ()
- virtual void **encode** (const vector< uchar > &x, vector< uchar > &y)
- virtual void **encode** (const uchar ∗inp, const unsigned inlen, uchar ∗outp, const unsigned outlen)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/enc_dec.h
- /home/kapil/libmimo/lib/enc_dec.cc

3.10 mimo::mod_demod Class Reference

Inheritance diagram for mimo::mod_demod:



Public Member Functions

- **mod_demod** (const vector< complex< float > > &constellation)
- int **mod_order** ()
- unsigned **bits_per_symbol** ()
- virtual vector< complex< float > > **points** ()

Protected Member Functions

- void **set_bps** ()

Protected Attributes

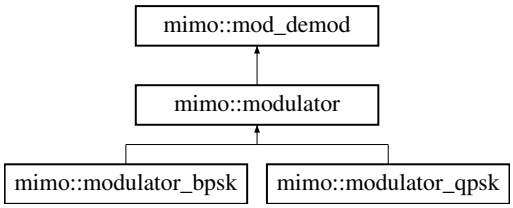
- int **d_mod_order**
- unsigned **d_bps**
- vector< complex< float > > **d_points**

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.11 mimo::modulator Class Reference

Inheritance diagram for mimo::modulator:



Public Member Functions

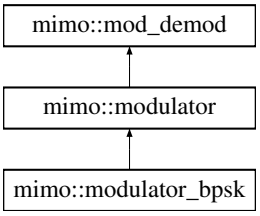
- **modulator** (const vector< complex< float > > &constellation)
- virtual complex< float > **encode** (const unsigned idx)
- virtual void **encode** (const unsigned idx, complex< float > *out)
- virtual void **encode** (const unsigned *idx, complex< float > *out, const unsigned vlen)
- virtual void **encode** (const mimo::uchar *idx, complex< float > *out, const unsigned vlen)
- virtual void **encode** (const vector< unsigned > &idx, vector< complex< float > > &out)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.12 mimo::modulator_bpsk Class Reference

Inheritance diagram for mimo::modulator_bpsk:



Public Member Functions

modulator_bpsk - BPSK modulator class

Returns

modulator_bpsk object

- **modulator_bpsk** ()

~modulator_bpsk - default destructor

Returns

void

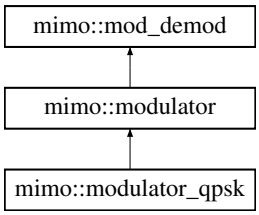
- ~**modulator_bpsk** ()

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.13 mimo::modulator_qpsk Class Reference

Inheritance diagram for mimo::modulator_qpsk:



Public Member Functions

modulator_qpsk - Grey coded QPSK modulator class

Returns

modulator_qpsk object

- **modulator_qpsk** ()

~**modulator_qpsk - default destructor**

Returns

void

- ~**modulator_qpsk** ()

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/mod_demod.h
- /home/kapil/libmimo/lib/mod_demod.cc

3.14 mimo::repacker Class Reference

Public Member Functions

- **repacker** (const unsigned k, const unsigned l)
- void **operator()** (const uchar ∗inp, const unsigned inlen, uchar ∗outp, const unsigned outlen)

The documentation for this class was generated from the following files:

- /home/kapil/libmimo/include/repacker.h
- /home/kapil/libmimo/lib/repacker.cc